

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Inhaltsverzeichnis

Einleitung.....	2
Vorgehensweise.....	2
Aufbau des Systems.....	2
Erläuterung des Grundsystems.....	2
Systemarchitektur.....	3
Anwendungsarchitektur.....	5
Entwurfsmuster und Architekturentscheidungen.....	5
Schichten der Systemarchitektur.....	6
Persistenzentscheidungen.....	8
Vorgegebene Geschäftsregeln.....	9
Integration von bestehenden Systemen.....	11
Verteilung des Systems.....	11
Kommunikation zwischen Controller und Simulation.....	12
Ausnahmebehandlung.....	13
Interaktion mit der Simulation durch Benutzer.....	14
Grafische Benutzeroberfläche.....	15
Aufbaue der Oberfläche, wie wird sie geschrieben?.....	15
Wireframe.....	15
Optionsmenü.....	16
Bearbeiten der Konfiguration.....	17
Screendesign.....	18
Anmerkungen zum Screendesign:.....	21

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Einleitung

In diesem Dokument wird beschrieben, was der tatsächliche Umfang des Projektes „Simulation des Transport der FDZ“ ist. Darunter fallen die Punkte, die mit den Auftraggebern aufbauend auf der Anforderungsanalyse (Lastenhefts) besprochen und angeglichen wurden. Das Ziel dieses Pflichtenheftes ist es, eine eindeutige Aussage über den Umfang des Projektes zu schaffen, dem beide Parteien, Entwicklerteam und Auftraggeber, zustimmen.

## Vorgehensweise

Das Entwicklerteam trifft sich wöchentlich mit dem Auftraggeber zu einem kurzen Scrum-Meeting. Der Auftraggeber gibt eine grobe Aufgabenstellung, die vom Entwicklerteam in konkrete Einträge im Produkt Backlog (siehe Backlog.ods) umgewandelt werden, diese werden anschließend vom Auftraggeber Priorisiert. Diese Priorisierung wird bei der Planung eines Sprints berücksichtigt. Die Zuteilung der Aufgaben an einzelne Teammitglieder geschieht, wie in Scrum vorgesehen, auf Wunsch der Mitglieder selbst.

## Aufbau des Systems

In diesem Kapitel wird beschrieben, wie das System aufgebaut ist. Dabei wird auf die Architektur des Systems, aber auch auf die Softwarearchitektur eingegangen.

## Erläuterung des Grundsystems

Die Fabrik der Zukunft besteht aus mehreren Teilbereichen, einer davon ist der Transport. Der Transport selber wird in mehrere Schichten unterteilt, für uns relevant ist die Darstellungsschicht des Transportes und der Controller des Transportes. In der Hierarchie des Systems liegt der Controller über der Darstellung, seine für dieses Projekt relevante

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

Aufgaben sind es, der Darstellungsschicht die Befehle zuzuschicken, die der Transportteil der Fabrik aktuell erledigt.

Diese Befehle sind in einem eigenen Protokoll beschrieben (siehe DocumentationNetworkCommands.pdf Kapitel 4.1) und werden per TCP/IP über eine Netzwerkschnittstelle an den PC, auf dem die Darstellung läuft, geschickt.

Die Darstellungsschicht ist eine Simulation, welche in einem Java-Programm geschrieben wird. Dies gehört zum Umfang dieses Projektes. Diese Simulation empfängt die TCP/IP Packets, sendet eine Rückmeldung, dass es diese empfangen hat, interpretiert sie und sendet erneut eine Nachricht an den Controller, dass die Nachricht entweder erfolgreich abgearbeitet wurde oder ein Fehler aufgetreten ist (Zur Netzwerkkommunikation später mehr).

Die Hardwaregrundlage des Simulations-Programmes bildet ein Standard Rechner, auf dem Java 1.8 installiert und lauffähig ist. In dieser Version wird das Programm auch entwickelt. Das zugrundeliegende Betriebssystem ist irrelevant, da das Programm in der Java-Virtual-Machine läuft und somit portierbar auf alle Java-fähigen Systeme ist. Einige Voraussetzung an den Rechner ist, dass dieser eine funktionierende Netzwerkschnittstelle hat, welche im Netzwerk der Fabrik der Zukunft angebunden ist.

## Systemarchitektur

Wie schon beschrieben wird die Anwendung der Simulation in der Programmiersprache Java entwickelt. Die Entscheidung hierfür wurde vom Auftraggeber und den Zuständigen für die Fabrik der Zukunft übernommen. Einige Gründe hierfür sind folgende:

- Portierbarkeit der Anwendung auf alle Rechner, die Java-fähig sind.
- Programmiersprache, die bei der Entwicklung der Fabrik der Zukunft eine große Rolle spielt.

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

- Eigenes Framework für Grafische Benutzeroberflächen (JavaFX).
- Einfache Anbindung des Systems an die Netzwerkschnittstelle

Die genaue Version von Java, die genutzt werden soll, ist die Version Java 1.8, die Wahl wurde getroffen, weil der Rest der auf Java aufbauenden Programme in der Fabrik der Zukunft ebenfalls in dieser Version entwickelt wurden und da auf den Rechnern, die in den Laboren der Fabrik der Zukunft zur Verfügung stehen, großenteils noch die Version 1.8 statt ihrer Nachfolger installiert ist.

Die Grafische Benutzeroberfläche, welche einen sehr wichtigen Teil des Systems bildet, wird in JavaFX entwickelt. Dies hat einige klare Vorteile:

- JavaFX ist das Java-Framework für GUIs und ist somit ohne weiteres an die Logik anbindbar.
- Die Trennung nach Model View Controller ist durch das Nutzen von FXML-Dateien und des Schreibens von Controllern für diese sehr gut möglich. So ist die unabhängige Entwicklung von Model und View sehr einfach.
- Die Entwicklung einer Oberfläche nach den Wünschen des Kunden durch den Scenebuilder für JavaFX ist sehr einfach und zügig erledigt.

Die Wahl der richtigen Protokolle für die Netzwerkkommunikation ist durch den Auftraggeber auf TCP/IP und das eigene Kommunikationsprotokoll der Fabrik der Zukunft gefallen.

TCP/IP wurde aus dem Grund gewählt, da die Übertragung der IP-Packets durch das Transmission-Control-Protocol garantiert wird. Solange also eine Netzwerkverbindung zwischen der Simulationsschicht und dem Controller des Transportes steht, so ist die

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

vollständige Übertragung der Nachrichten auch bei Netzwerküberlastung garantiert. Dieses Konzept wurde durchgängig in allen Systemen der Fabrik der Zukunft gewählt.

Genauso wurde auch das eigene Kommunikationsprotokoll konsistent genutzt. Dieses hat den Vorteil, dass seine Befehle speziell auf die Befehle der Fabrik abgestimmt sind (siehe DocumentationNetworkCommands.pdf Kapitel 4.1). Es können im Verlauf der Verfeinerung der Systeme sogar noch weitere Befehle eingefügt werden, was aber in diesem Projekt zur Entwicklung der Simulation nicht erwünscht ist.

## Anwendungsarchitektur

In diesem Kapitel wird nun detailliert auf den Aufbau der Anwendungsarchitektur eingegangen. Behandelt werden hier unter Anderem, welche Designentscheidungen das Team in Absprache mit den Auftraggebern getroffen hat und welche Entwurfsmuster für die Entwicklung genutzt werden. Anschauliche Beispiele werden auch gezeigt.

**Anmerkung:** Dieses Projekt wird agil durch die Vorgehensweise Scrum durchgeführt. Zu beachten ist deshalb, dass das Team sich im Vornherein nur das Grobkonzept der Architektur herleiten kann. Dennoch sind früh im Projekt einige Leitfäden entworfen worden, nach denen sich die Entwicklern richten müssen.

## Entwurfsmuster und Architekturentscheidungen

Nach Absprache mit dem Auftraggeber ist es das große Ziel dieses Projektes, für die Simulation ein großes Maß an Wiederverwendbarkeit und Modularität zu gewährleisten. Ein gewählter Schritt, um die Wiederverwendbarkeit der Grundarchitektur zu garantieren ist die Entwicklung des System durch das Entwurfsmuster Model-View-Controller. Dies ist nicht nur guter Stil in der Programmierung, es erlaubt dem Team auch, die gesamte Logik des Systems im Voraus zu implementieren und zu testen. Nach der Fertigstellung der Grundlage kann dann mit dem JavaFX Scenebuilder eine Grafische Oberfläche nach den Wünschen der Auftraggeber (dazu in späteren Kapiteln mehr) entworfen werden, die dann

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

nur noch durch einem Controller mit der Logik verknüpft werden muss. Die Vorteile liegen auf der Hand, die Logik kann jederzeit abgekoppelt und an eine neue Oberfläche angebunden werden. Der Programmiercode wird somit auch besser wiederverwendbar und lesbarer.

## Schichten der Systemarchitektur

Im folgenden wird schematisch dargestellt, in welche Schichten das System aufgeteilt wird:

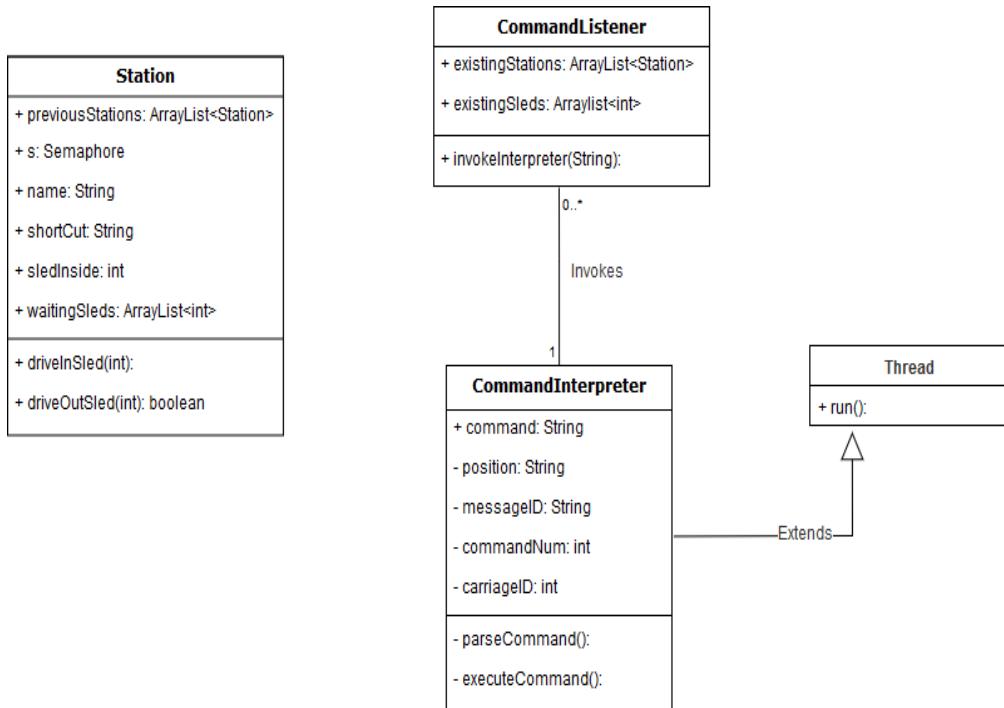


Abbildung 1: Klassendiagramm

**Abbildung 1:** Der **CommandListener** ist dafür zuständig Kommandos über die Netzwerkschnittstelle zu empfangen, einen **CommandInterpreter** zu erzeugen der dieses interpretiert und die Antwort des Interpreters zurückzusenden. Der **CommandInterpreter** erbt von der Klasse **Thread** so können mehrere Kommandos unabhängig voneinander gleichzeitig bearbeitet werden. Das erhaltene Kommando wird vom **CommandInterpreter**

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

geparst, in die Bestandteile aufgeteilt und Ausgeführt. Die Klasse Station verwaltet eine Station, dazu gehört der Name der Station, die Abkürzung, die direkten Vorgänger der Station und welche Schlitten an ihr anstehen. Die driveIn und driveOut Methoden einer Station sollen durch ein Semaphor vor gleichzeitigen Zugriffen geschützt werden, dies wird in Java durch die Deklaration der Methoden als synchronised umgesetzt.

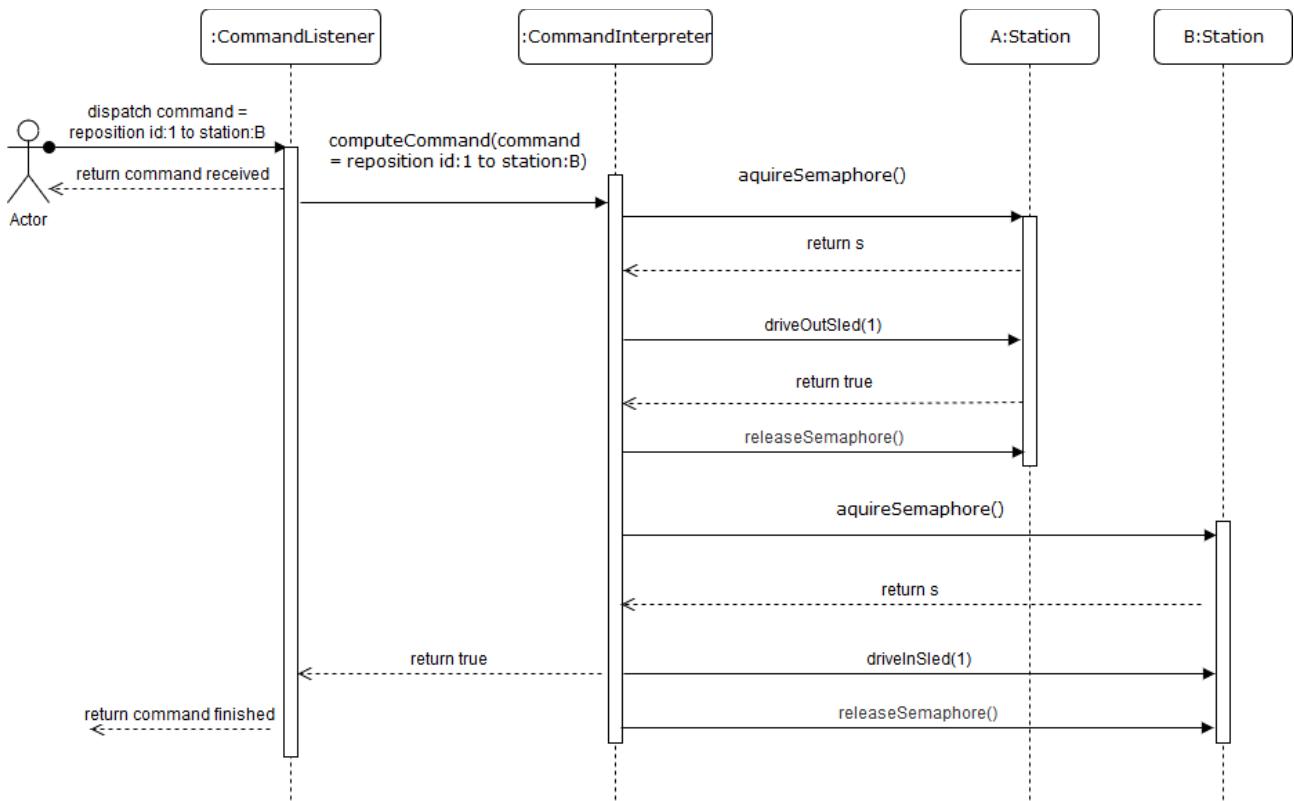


Abbildung 2: Sequenzdiagramm

## Begründungen

Die Entscheidung für die Erstellung verschiedener Klassen für die Interpretation und das Empfangen der Kommandos folgert sich aus der Forderung, theoretisch unendlich viele Kommandos auf einmal bearbeiten zu können. Deshalb wird der Kommando Interpreter

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

auch als Subklasse von Thread implementiert. Bei dem Empfangen eines Befehls wird eine neue Instanz des Interpreters erstellt, welche dann in einem eigenen Thread weiterarbeitet, der unabhängig vom restlichen System läuft. So können alle Befehle parallel bearbeitet werden, sobald sie empfangen wurden. Die zu schützenden Daten, die von den Interpreter-Threads parallel manipuliert werden, sind die Stationen. Diese werden beim Zugriff von außen mit einem binären Semaphore geschützt.

## Persistenzentscheidungen

Zum Umfang des Projektes gehört eine Erstellung einer Log-Datei, die die Vorgänge im System und die eingegangenen Nachrichten der Netzwerkschnittstelle aufnimmt und entsprechend abspeichert. Die Log-Datei hat folgende Anforderungen zu erfüllen:

- Erstellen einer Datei mit dem aktuellen Datum im Namen bei Systemstart.
- Sammeln aller Netzwerknachrichten und persistentes Speichern in menschenlesbarem Fließtext, sobald diese eingetroffen sind.
- Sammeln aller Fehler, die mit einer Nachricht in Verbindung stehen.
- Sind 500 Zeilen an Einträgen in der Datei, muss eine neue Datei geöffnet werden. In der neuen Datei ist zu vermerken, dass diese eine Fortführung der vorherigen ist.
- Sollte das System um 23:59Uhr eines Tages laufen, so wird um Punkt 0:00Uhr eine neue Log-Datei mit neuem Datum angefangen. In der neuen Datei muss vermerkt sein, dass es eine Fortführung des vorherigen Tages ist.
- Sollte das System an einem Tag mehrmals gestartet werden, so muss die Log-Datei des Tages weiterverwendet werden, bis diese das Zeilenlimit erreicht hat.

Eine weitere Anforderung ist eine Datei im JSON-Format, die den Zustand der Simulation abspeichert. Diese Datei muss folgende Anforderungen erfüllen:

- Die Datei wird beim Systemstart angelegt und immer aktualisiert, sobald sich der Zustand der Simulation ändert.

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

- Bei Absturz des Systems muss es möglich sein, den Zustand vor dem Absturz aus der Datei wiederherzustellen. Es wird dem Anwender die Frage gestellt, ob er den Zustand wiederherstellen möchte oder nicht. Diese Funktion soll der Anwender nur nach einem Absturz haben, nie nach einem kontrolliertem Herunterfahren.
- Bei korrektem Herunterfahren der Simulation soll die Datei gelöscht werden. So wird bei einem Neustart des Systems gesucht, ob die Datei vorhanden ist. Ist dies nicht der Fall, so muss der Zustand auch nicht wiederhergestellt werden.

Die genauen Entscheidungen, wie das Speichern durch Java-Streams realisiert wird, kann erst zum Zeitpunkt des Entwickelns getroffen werden.

Im folgenden werden einige erläuternde Diagramme zur Verdeutlichung des Speichervorgangs der Dateien angegeben:

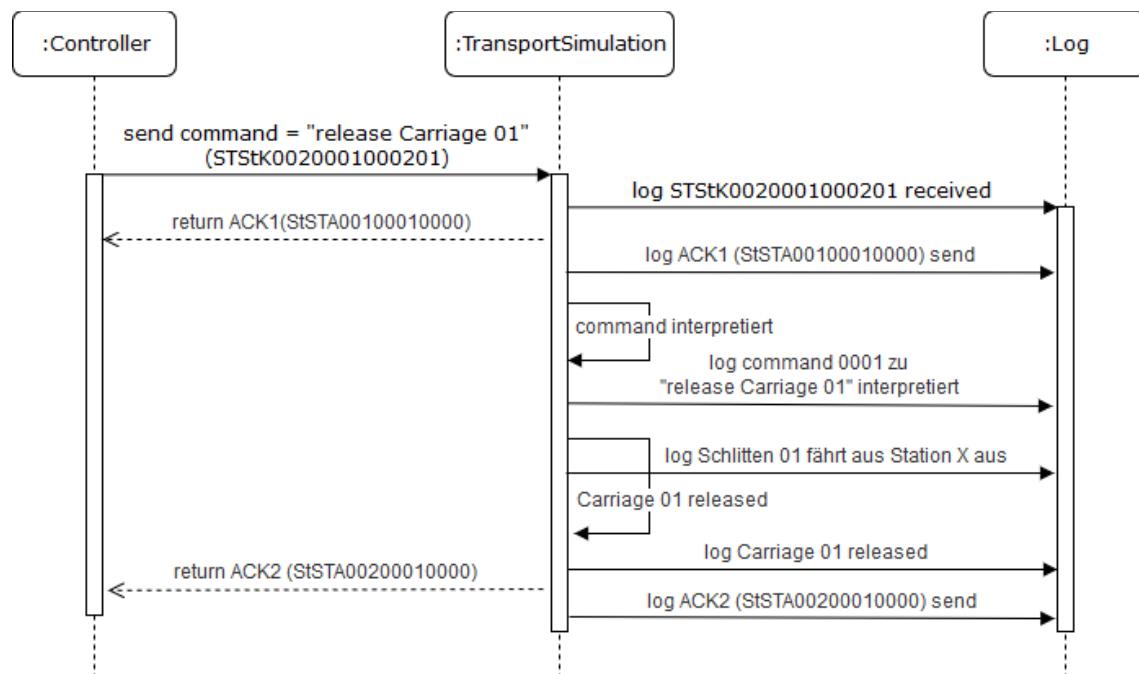


Abbildung 3: Log Sequenzdiagramm

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Vorgegebene Geschäftsregeln

Aus den Gesprächen mit den Auftraggebern haben sich folgende Punkte als essentiell herausgestellt:

- Netzwerkkommunikation:

Die Abläufe für die Kommunikation mit dem übergeordneten Controller sind wie folgt gegliedert: (Genaue Befehle in Doku FDZ)

1. Controller sendet Nachricht an Simulationsschicht
2. Simulationsschicht empfängt Nachricht und sendet eine Bestätigungs Nachricht (Acknowledge) an den Controller, dass die Nachricht angekommen ist.
3. Controller wartet auf weitere Nachricht, Simulation arbeitet initiale Nachricht ab.
4. Simulationsschicht sendet Nachricht über den Ausgang der Abarbeitung an den Controller, entweder eine Betätigung oder eine Fehlermeldung.

- Netzwerkverbindung:

Die Reaktion auf Probleme der Verbindung muss genau beschrieben und danach eingehalten werden. Probleme, die getestet und abgefangen werden müssen, sind folgende:

- Verbindung bricht ab, ohne, dass aktuell Nachrichten ausgetauscht werden.
- Verbindung bricht ab, nachdem der Controller die Nachricht mit dem Befehl an die Simulation geschickt hat und diese empfangen wurde.
- Verbindung bricht ab, nachdem Simulation Empfangsbestätigung an Controller geschickt hat.
- Erweiterung der Stationen

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

Das System der Fabrik der Zukunft ist aufgeteilt auf ein Transportband und mehrere Stationen. Zum Zeitpunkt der Entwicklung sind es genau 3 Stationen:

- Roboter (RO)
- Lager (LA)
- Ein-/Ausgabe (EA)

Es soll aber in Zukunft möglich sein, neue Stationen zu simulieren, wenn diese in die Fabrik übernommen werden. Deshalb wurde sich darauf geeinigt, dass die Stationen in Form einer verketteten Liste abgespeichert werden, in die neue Knoten eingefügt und auch entfernt werden können. Dies soll über eine einfache Schnittstelle in der grafischen Oberfläche möglich sein, ohne etwas im Source-Code umschreiben zu müssen. Das Design hierfür wird später genauer betrachtet.

- Ablauf des Transportes:

Eine große Fehlerquelle im System kann es sein, dass der Transportweg eines Güterobjekts durch ein anderes Güterobjekt versperrt ist. Darauf wird folgendermaßen reagiert:

- Das System erkennt, dass sich Güter auf dem Transportband aufstauen
- In der Grafischen Oberfläche ändert sich ein Wert, der symbolisiert, dass ein Stau vorliegt.

Da eine Reaktion auf den Stau nicht möglich ist, wird dieser nur angezeigt.

- Reaktion auf Nachrichten des Controllers:

Da es in Zukunft möglich ist, neue Stationen einzufügen, muss aus den Nachrichten des Controllers immer erst überprüft werden, ob diese Station tatsächlich im System der Transportsimulation vorhanden ist.

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Integration von bestehenden Systemen

Die Entwicklung der Simulation des Transportes kann weitläufig unabhängig erfolgen. Die einzigen Bedenken, die der Integration von vorhandenen Systemen gewidmet werden müssen, ist die mehrmals erwähnte Netzwerkkommunikation. Das schon vorhandene System ist hier der Controller des Transportes. Da auf die Funktion der Netzwerkkommunikation und der zugrundeliegenden Protokolle schon mehrmals eingegangen wurde, wird das in diesem Kapitel nicht erneut aufgegriffen.

## Verteilung des Systems

Im allgemeinen ist das System der Fabrik der Zukunft auf einige verschiedene untergeordnete Systeme verteilt. Im Transport spiegelt sich das auch wieder. Für dieses Projekt relevant sind allerdings nur 2 Untersysteme, der Controller und die Simulation des Transportes. Diese kommunizieren ausschließlich über eine Netzwerkschnittstelle, es Bedarf hier also keiner weiteren Anbindung. Die Simulation an sich läuft auf einem einzelnen Rechner, der nach außen nur über die Netzwerkschnittstelle für andere Untersysteme erreichbar ist. Demnach muss sich das Entwicklerteam im Rahmen dieses Projektes keine weiteren Gedanken über die Verteilung des Systems machen.

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Kommunikation zwischen Controller und Simulation

Es wurde im Kapitel *Vorgegebene Geschäftsregeln* schon eingehend auf den Ablauf der Kommunikation eingegangen, deshalb wird hier nur noch ein erläuterndes Sequenzdiagramm angegeben:

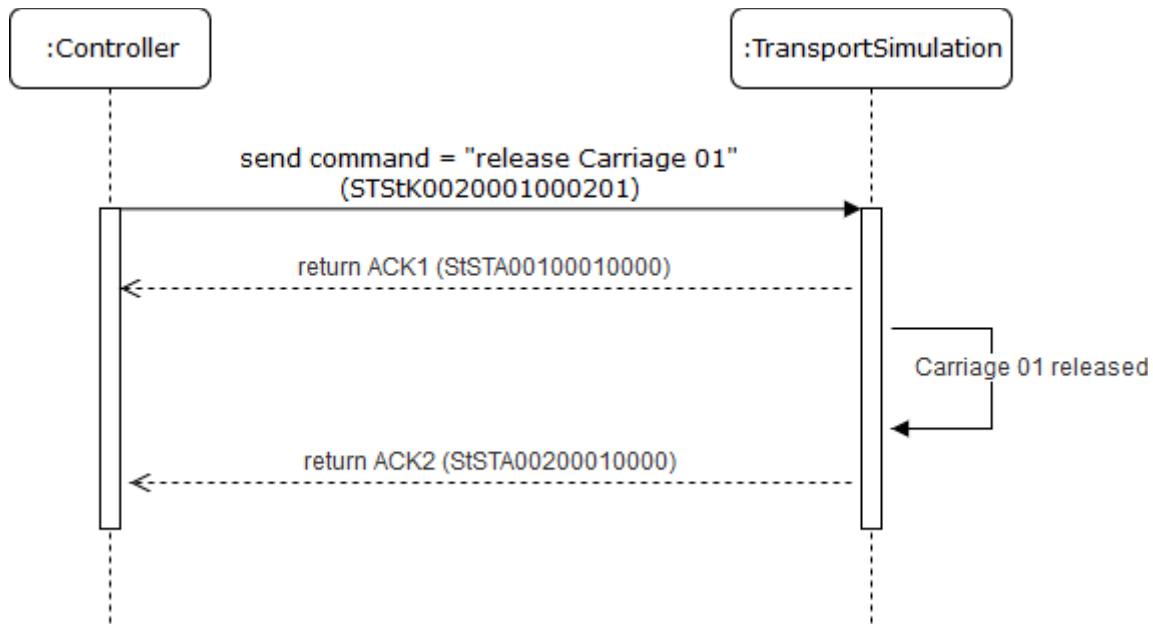


Abbildung 4: Netzwerk Sequenzdiagramm

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Ausnahmebehandlung

Im groben kann gesagt werden, dass das Fehlerabfangen, wie in Java typisch, durch ExceptionHandling realisiert wird. Einige häufige Fehler, die getestet und abgefangen werden müssen, sind folgende:

- Fehler in Netzwerkkommunikation, Abbruch der Verbindung:

Bei diesem Fehler wird eine Exception geworfen, die alle weiteren Mechanismen ihre aktuelle Aufgabe zuende arbeiten lässt, und danach alles zum Stillstand bringt.

Die Exception muss dementsprechend behandelt werden, indem die Verbindung wieder aufgebaut wird und dem Benutzer eine Meldung angezeigt wird, dass die Verbindung abgebrochen ist.

- Nicht erkannte Station:

Die über die Netzwerkschnittstelle empfangenen Befehle bearbeiten großenteils die Zustände, in denen sich die Paletten, mit oder ohne Inhalt, oder die Stationen befinden.

Sollte ein Befehl eine Palette oder Station ansprechen, die dem System nicht bekannt ist, dann muss eine Fehlermeldung an dem Controller weitergeleitet werden.

- Unbekannter Befehl empfangen:

Die Befehle, die der Controller schickt, sollten zur Programmierphase bekannt sein.

Sollte nun ein unbekannter Befehl empfangen werden, so wird dem Controller dies in einer Fehlermeldung mitgeteilt.

- Stau an einer Station:

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

Eine Station der Fabrik der Zukunft kann nur genau eine Palette bearbeiten, das System ist aber so konzipiert, dass es theoretisch unendlich viele Paletten und Befehle gleichzeitig abarbeiten kann.

Sollte es nun dazu kommen, dass das Weiterarbeiten an einer Palette durch einen Stau an einer Station nicht möglich ist, so muss dies in der betroffenen Station angezeigt werden.

## Interaktion mit der Simulation durch Benutzer

Wie schon eingehend erwähnt ist die Hauptaufgabe der Simulation einem Nutzer der Anwendung den Ablauf des Transportes optisch darzustellen. Deshalb wird dem System eine grafische Benutzeroberfläche angefügt, nachdem die Logik implementiert wurde. Die Anforderungen der Auftraggeber sind folgende:

- Einfache Bedienung der Oberfläche mit Maus und Tastatur des Rechners, auf dem die Anwendung läuft.
- Intuitive Navigation des Systems durch sehr einfache Funktionen, klaren Darstellungen von interaktiven Feldern und Auslagerung von Optionen in ein Menü.
- Darstellung der Änderungen in der Oberfläche in zwei verschiedenen Modi, in Rechenzeit, also nicht lesbar für das menschliche Auge, oder in Echtzeit, also so, wie das Transportband die Änderungen abarbeitet. Diese Zeiten sollen im Menü abänderbar sein.

Weitere Informationen zu der Oberfläche, der Usability und der Designentscheidungen werden im folgenden Kapitel erörtert und beispielhaft dargestellt.

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Grafische Benutzeroberfläche

Dieses Kapitel beschreibt Aufbau und Funktionsweise der Grafischen Benutzeroberfläche genauer.

### Aufbaue der Oberfläche, wie wird sie geschrieben?

Die Oberfläche ist in eine FXML-Datei und mehrere JavaFX Klassen unterteilt. Die FXML-Datei stellt die View der Anwendung dar und beschreibt das Grundlegende Aussehen der Benutzeroberfläche, sie kann über den JavaFX Scenebuilder leicht, auch von nicht Informatikern, bearbeitet werden. Die JavaFX Klassen hingegen stellen den Controller dar, sie sind zuständig für Benutzer bedingte Veränderungen in View und Model und Kommando bedingte Änderungen in der View.

### Wireframe

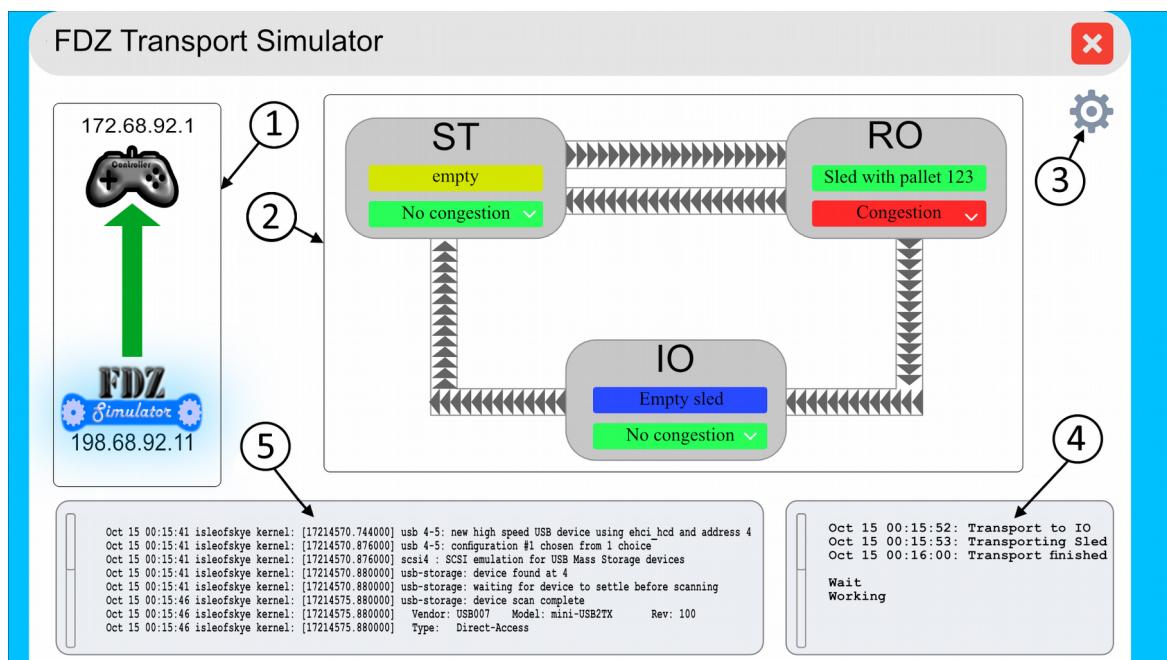


Abbildung 5: Wireframe

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

**Abbildung 5:** Der Nutzer sieht im linken Bereich der Oberfläche (1) ob und über welche IP-Adressen eine Verbindung zu dem übergeordneten Controller und dem untergeordneten Physikalischen Fließband besteht. Im Zentrum der Oberfläche (2) ist die, vom Nutzer veränderbare, Konfiguration der Stationen und Fließbänder zu sehen. Die Stationen zeigen an welche Schlitten mit welchen Paletten-IDs bei ihnen anstehen und ob momentan ein Stau vorhanden ist. Oben rechts (3) ist ein Zahnrad zu sehen, über welches der Nutzer in das Optionsmenü gelangt, dessen Optionen werden im Folgenden behandelt. Unten rechts (4) ist die scrollbare Auflistung der Momentanen Tätigkeit und vergangener Tätigkeiten. Unten links (5) ist der Log zu sehen, dieser ist identisch mit dem der Log Dateien, zeigt also eingehende und ausgehende Netzwerk Nachrichten und Fehlermeldungen an.

## Optionsmenü

Folgende Optionen sollen implementiert werden:

- Echtzeit-Modus: bei Aktivierung wird eine künstliche Verzögerung beim Verarbeiten der Kommandos eingefügt um die Geschwindigkeit des echten Systems zu simulieren
- Station hinzufügen: bei Betätigung wird eine neue Station der Konfiguration hinzugefügt, diese muss noch Benannt und mit den anderen Stationen verbunden werden, um einsatzbereit zu sein
- Konfiguration bearbeiten: bei Aktivierung sind die Stationen verschiebbar, ihre Verbindungen zu anderen Stationen sowie ihr Name können verändert werden und sie können gelöscht werden und die Bahn der Fließbänder kann verändert werden

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

## Bearbeiten der Konfiguration

Bei Aktivierung der „Konfiguration bearbeiten“ Option können die Stationen per Drag&Drop verschoben werden. Weiterhin haben die Stationen nun ein Zahnrad über welches sich das Konfigurationsmenü der einzelnen Station öffnen lässt. Über das Konfigurationsmenü lässt sich eine Station löschen, ihr Name verändern und die Liste der Stationen bearbeiten die diese Station direkt erreichen können, auf Grundlage dieser Liste werden die Fließbänder eingezeichnet.

Außerdem kann die Bahn von Fließbändern verändert werden. Indem man auf ein Fließband doppelklickt entsteht dort ein Eckpunkt (siehe Abbildung 6), der sich per Drag&Drop verschieben lässt. Diese Eckpunkte können wieder entfernt werden indem man auf sie doppelklickt.

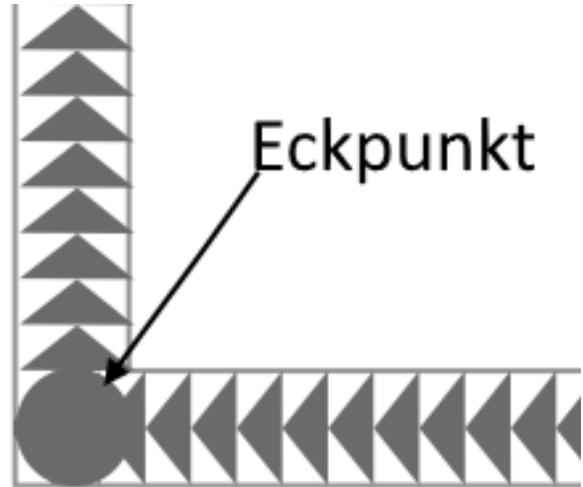
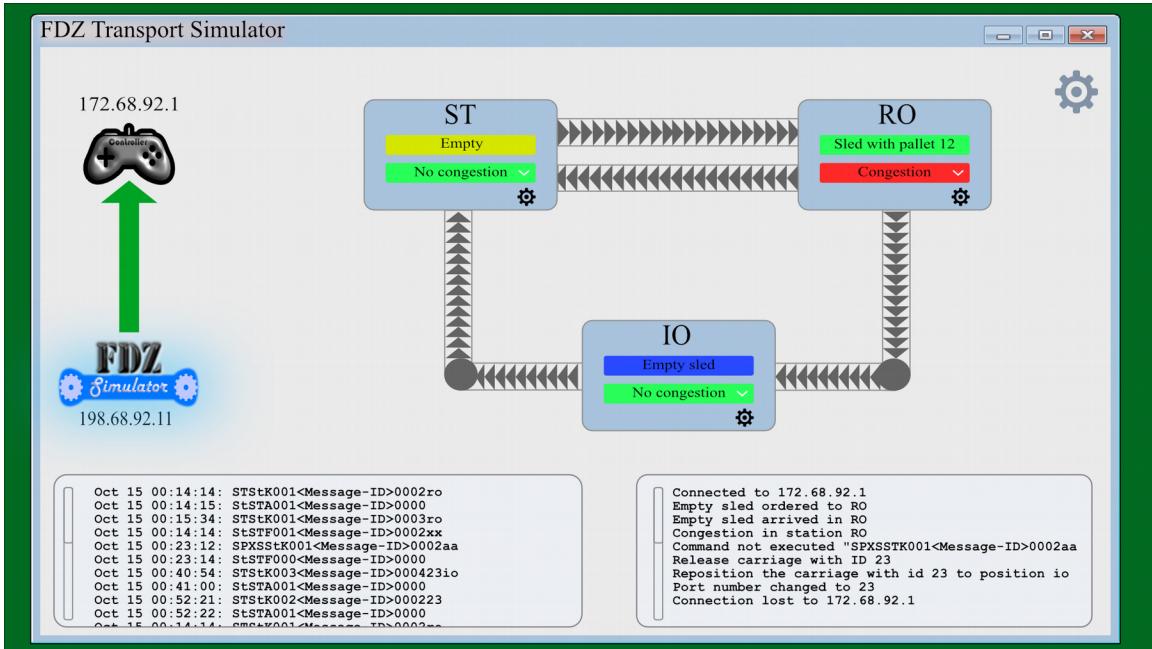


Abbildung 6: Eckpunkt

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

# Screendesign



*Abbildung 7: Screendesign*

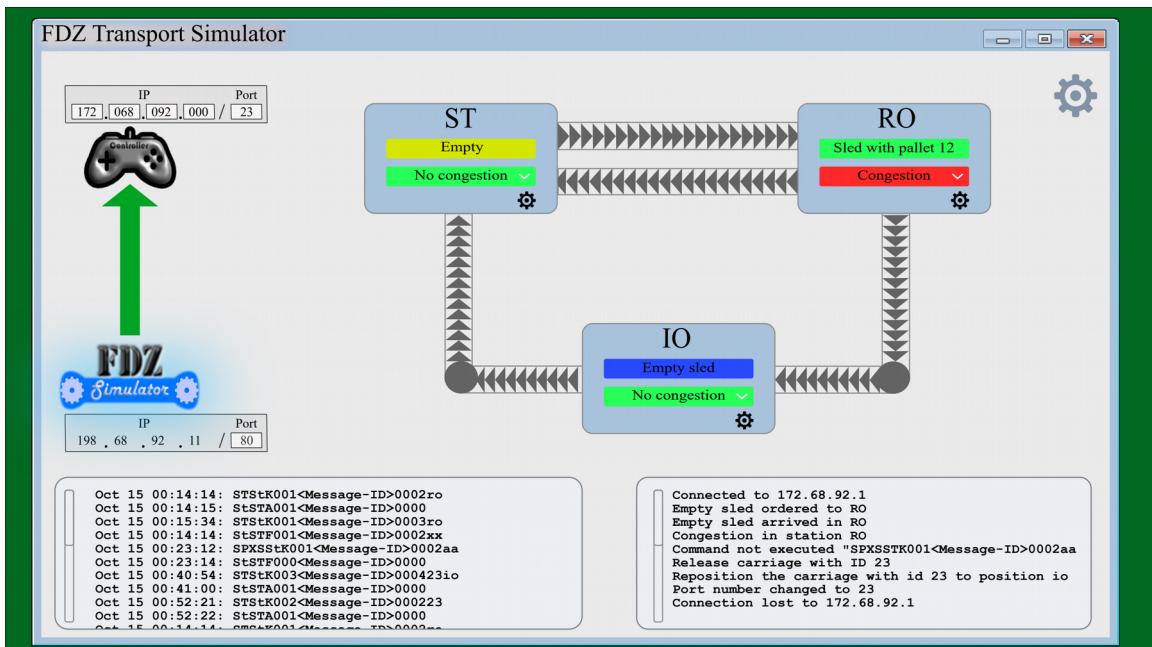


Abbildung 8: Screendesign (Ändern der IP-Konfiguration)

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

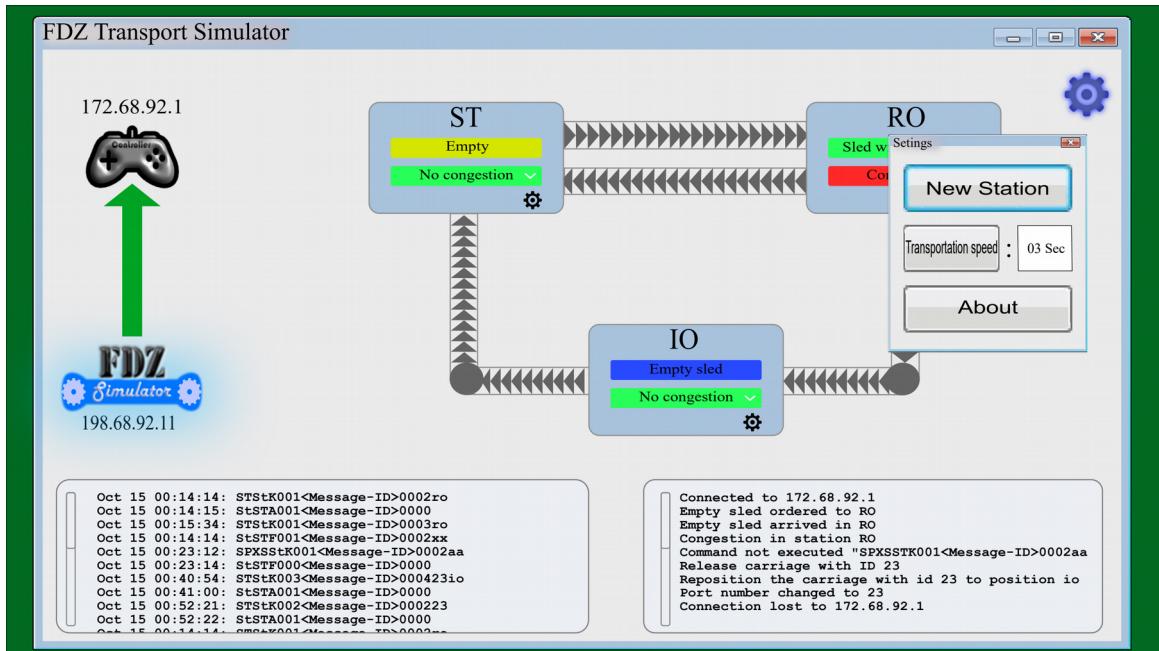


Abbildung 9: Screendesign (Optionen)

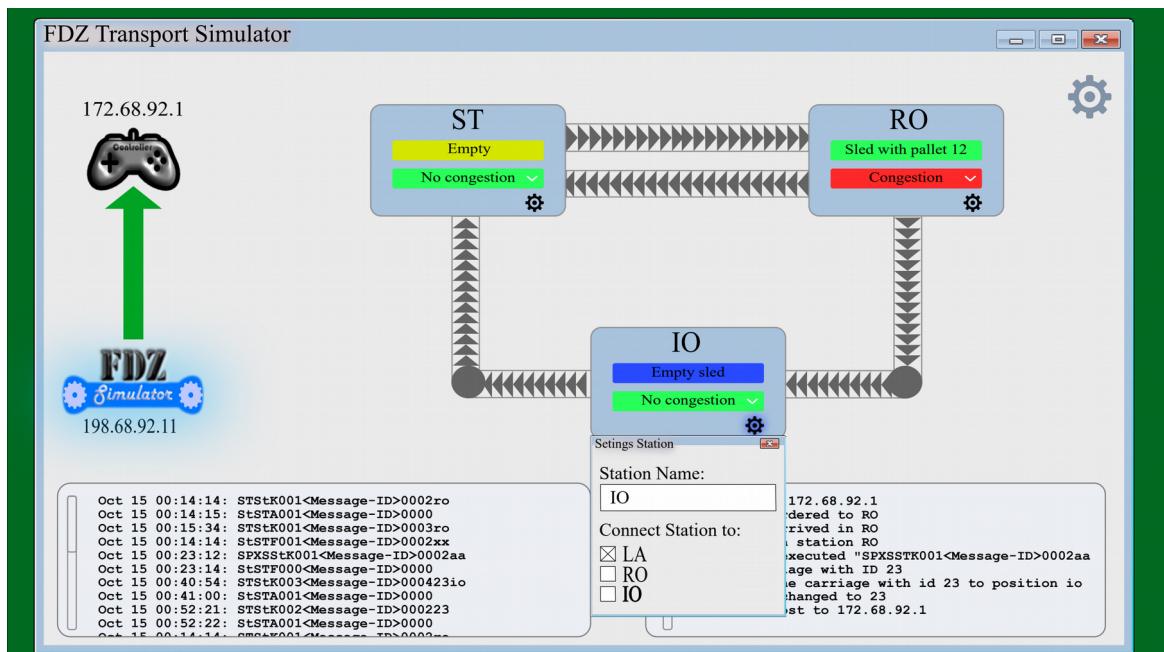


Abbildung 10: Screendesign (Stationen Konfigurieren)

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

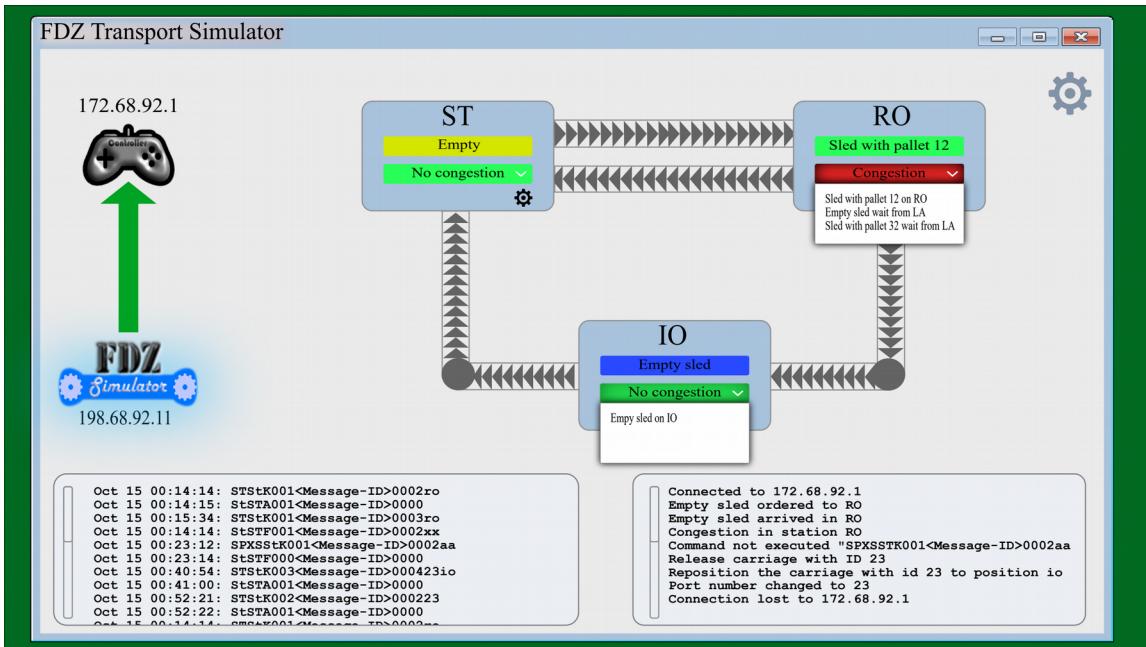


Abbildung 11: Screendesign (Werte in Station)

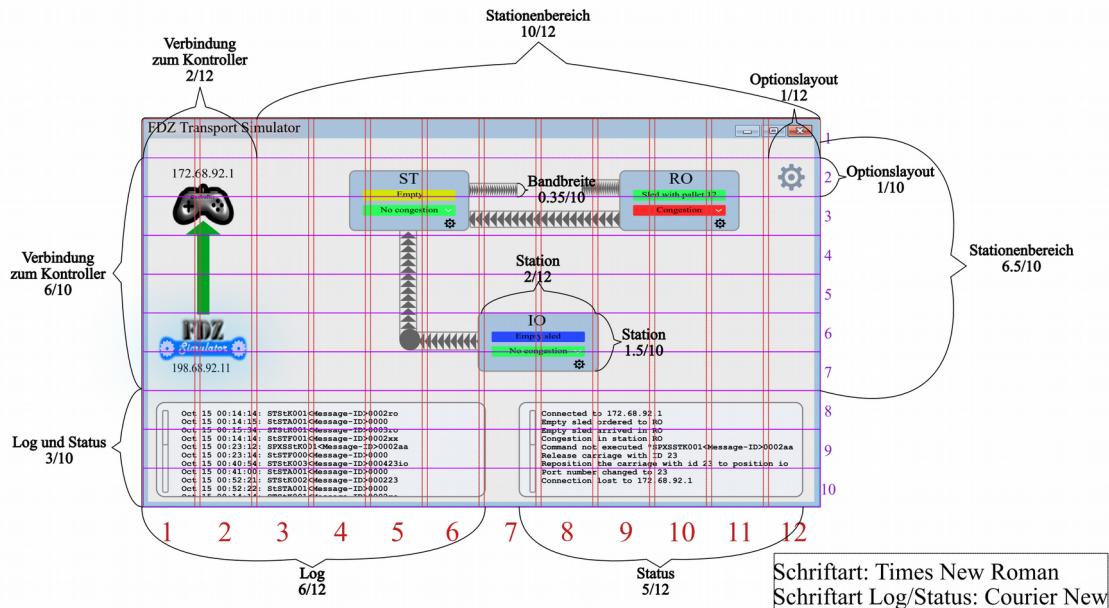


Abbildung 12: Screendesign (Seiten- und Größenverhältnisse)

Simulation FDZ	Praktikum Software Entwicklung
Pflichtenheft	17.04.2018

### Anmerkungen zum Screendesign:

Das Screendesign wurde anhand der Wireframes aus dem Lastenheft mit den Auftraggebern angefertigt. Das Screendesign basiert auf dem letzten Wireframe, welches auch hier im Pflichtenheft aufgeführt wurde. Die Anordnung und Größe der Stationen bleiben allerdings variabel, da im Laufe des Betriebes noch Stationen eingefügt und gelöscht werden können. Außerdem ist es möglich, die Anordnung der simulierten Fabrikelemente per Drag&Drop zu verändern.